# ИСПОЛЬЗОВАНИЕ GRADLE ДЛЯ СБОРКИ ПРОЕКТА

***Каторец А.Ф.,***
*студент 4 курса ВГУ имени П.М. Машерова, г. Витебск, Республика Беларусь*
Научный руководитель – Сергеенко С.В.

В современном мире все большую популярность приобретают web-приложения. Они не требовательны к ресурсам, мобильны, их не нужно устанавливать – для их использования нужен лишь браузер. При создании таких приложений используются различные платформы, технологии, методологии разработки, однако все они имеют что-то общее – процесс сборки.

Ни один крупный проект с использованием платформы Java не обходится без инструментов сборки. Собирать дистрибутив вручную не всегда удобно. А если в проекте используется несколько разных IDE, то лучше компилировать из консоли. Также перед сборкой дистрибутива нужно проставить номер версии в его имени. И unit-тесты запустить. А дальше – Continuous Integration. И CI сервер должен все это делать самостоятельно [1].

Для решения этих задач большинство разработчиков хоть раз, но использовали Ant или Maven. Есть и другие инструменты, Gradle – один из них.

Цель работы – изучить возможности инструмента Gradle для сборки проекта.

**Материал и методы.** Материалом исследования является инструмент для сборки Gradle. Методы исследования – анализ, сравнение.

**Результаты и их обсуждение.** Gradle – система автоматической сборки, построенная на принципах Apache Maven, основанного на концепции жизненного цикла проекта, и Apache Ant, в котором порядок выполнения задач определяется отношениями зависимости. Gradle использует направленный ациклический граф для определения порядка выполнения задач, а также вместо настройки с помощью XML, используется DSL Groovy [2].

Gradle был разработан для расширяемых много-проектных сборок, и поддерживает инкрементальные сборки, определяя, какие компоненты дерева сборки не изменились и какие задачи, зависимые от этих частей, не требуют перезапуска[3].

Достоинства Gradle по сравнению с Ant и Maven:

Использование ациклического графа для определения последовательности задач позволяет гибко настраивать Gradle для специфических нужд;

Возможность выполнять задачи Ant, а также возможно преобразовать pom.xml от Maven в скрипт для Gradle;

Множество способов для управления зависимостями – от удаленных Maven и Ivy[4] репозиториев с возможностью разрешения транзитивных зависимостей до локальных репозиториев;

Скрипт написан на языке Groovy, а не на XML, что облегчает его сопровождение и изменение, а также позволяет выделить логические части в скрипте, которые можно использовать повторно;

Wrapper – позволяет запустить скрипт без установки Gradle на компьютер.

Для выполнения сборки проекта можно использовать готовый сценарий. Для этого нужно подключить необходимый плагин. После подключения плагина можно выполнять задачи, описанные в плагине. Также, если это необходимо, можно создавать собственные задачи и включать их в готовые сценарии.

Для определения задач, которые нужно выполнять, Gradle проходит три отдельных фазы:

1. Инициализация – т.к. Gradle поддерживает одно- и многопроектные сборки, то на этом этапе определяется, какие проекты будут участвовать в сборке;

2. Конфигурация – во время этой фазы конфигурируются объекты проекта, а задачи собираются во внутреннюю объектную модель – направленный ациклический граф;

3. Выполнение – Gradle определяет подмножество задач, созданных и сконфигурированных на этапе конфигурации, для выполнения. Затем Gradle выполняет каждую из выбранных задач.

**Заключение.** Использование Gradle подразумевает написание скрипта сборки. В этом скрипте необходимо подключить плагины для задач, определить репозитории и зависимости проекта, а также, при необходимости, описать собственные задачи и сконфигурировать задачи плагинов. В результате мы получим простой и гибкий процесс сборки приложения.

Литература:

1. Gradle: Better Way To Build / Хабрахабр [Электронный ресурс]. – Режим доступа: https://habrahabr.ru/post/107085. Дата доступа: 09.02.2017.
2. Groovy DSL – A Simple Example – DZone Java [Электронный ресурс]. – Режим доступа: https://dzone.com/articles/groovy-dsl-simple-example. Дата доступа: 09.02.2017.
3. Gradle – Википедия [Электронный ресурс]. – Режим доступа: https://ru.wikipedia.org/wiki/Gradle. Дата доступа: 12.02.2017.
4. Apache Ivy – Национальная библиотека им. Н. Э. Баумана [Электронный ресурс]. – Режим доступа: http://ru.bmstu.wiki/Apache_Ivy. Дата доступа: 12.02.2017.
5. Gradle User Guide Version 3.4 [Электронный ресурс]. – Режим доступа: https://docs.gradle.org/current/userguide/tutorial_using_tasks.html. – Дата доступа: 17.02.2017.